REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. REPORT DATE *(DD-MM-YYYY)*	2. REPORT TYPE	3. DATES COVERED *(From - To)*
JUNE 2014	JOURNAL ARTICLE (Post Print)	APR 2012 – APR 2013

4. TITLE AND SUBTITLE

EVOLUTION OF EMBEDDED PROCESSING FOR WIDE AREA SURVEILLANCE

5a. CONTRACT NUMBER
IN-HOUSE

5b. GRANT NUMBER
NA

5c. PROGRAM ELEMENT NUMBER
62788F

6. AUTHOR(S)

C. Usmail, M. Little, R. E. Zuber

5d. PROJECT NUMBER
T2AW

5e. TASK NUMBER
IN

5f. WORK UNIT NUMBER
HO

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Air Force Research Laboratory/Information Directorate
Rome Research Site/RITB
525 Brooks Road
Rome NY 13441-4505

8. PERFORMING ORGANIZATION REPORT NUMBER
N/A

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
Air Force Research Laboratory/Information Directorate
Rome Research Site/RITB
525 Brooks Road
Rome NY 13441-4505

10. SPONSOR/MONITOR'S ACRONYM(S)
AFRL/RI

11. SPONSORING/MONITORING AGENCY REPORT NUMBER
AFRL-RI-RS-TP-2014-042

12. DISTRIBUTION AVAILABILITY STATEMENT
APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED. PA Case Number: 88ABW-2013-1738
DATE CLEARED: 11 APR 2013

13. SUPPLEMENTARY NOTES

14. ABSTRACT

Research and development being performed at the Air Force Research Laboratory's (AFRL's) Information Directorate is leading the way by providing a scalable, reconfigurable, embedded processing capability to ingest raw sensor data, process the data into information, and distribute the products to the war fighter. The airborne processing capability discussed in this article demonstrates substantial improvement over previously demonstrated capabilities. The heterogeneous AirWASP high-performance embedded computer (HPEC) system, Coeus, was developed using state-of-the-art, general-purpose graphical processing units (GPGPUs) and CPUs—all while considering stringent size, We review the AirWASP project and cover requirements, generation, processing needs, tradeoffs, system design, and our future vision.

15. SUBJECT TERMS
Embedded processing; high performance computing; general-purpose graphical processing units (GPGPUs)

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
					MICHAEL LITTLE
a. REPORT	b. ABSTRACT	c. THIS PAGE			19b. TELEPHONE NUMBER (*Include area code)*
U	U	U	UU	10	N/A

Evolution of Embedded Processing for Wide Area Surveillance

Courtney L. Usmail, Michael O. Little
Air Force Research Laboratory

Richard E. Zuber
SRC, Inc.

AESM 1XO is "Draft 1XO". Base alignment, orphans, widows, space above, space below will be addressed at 1st revision.
The **PINK FIGURE/TABLE LOCATOR TEXT** WILL RETURN TO BLACK AT 1REV

INTRODUCTION

Advancements in radar and sensor technology are essential for maintaining quality intelligence, surveillance, and reconnaissance (ISR) mission capabilities. The capabilities these advancements are achieving include the ability to provide persistent all-weather surveillance over wide areas of interest. With modern-day warfare, this technology is essential in tracking activity and change in complex urban environments where nonconventional tactics are employed [1]. One implication of the technological advancements is the creation of a massive data problem that requires processing to be both in real time and close to the sensor in order to disseminate the vital information products to war fighters to support and positively affect their mission. Significant improvements in high-performance computing (HPC) technology make it possible to enable such a data-to-decision information pathway in real or near-real time.

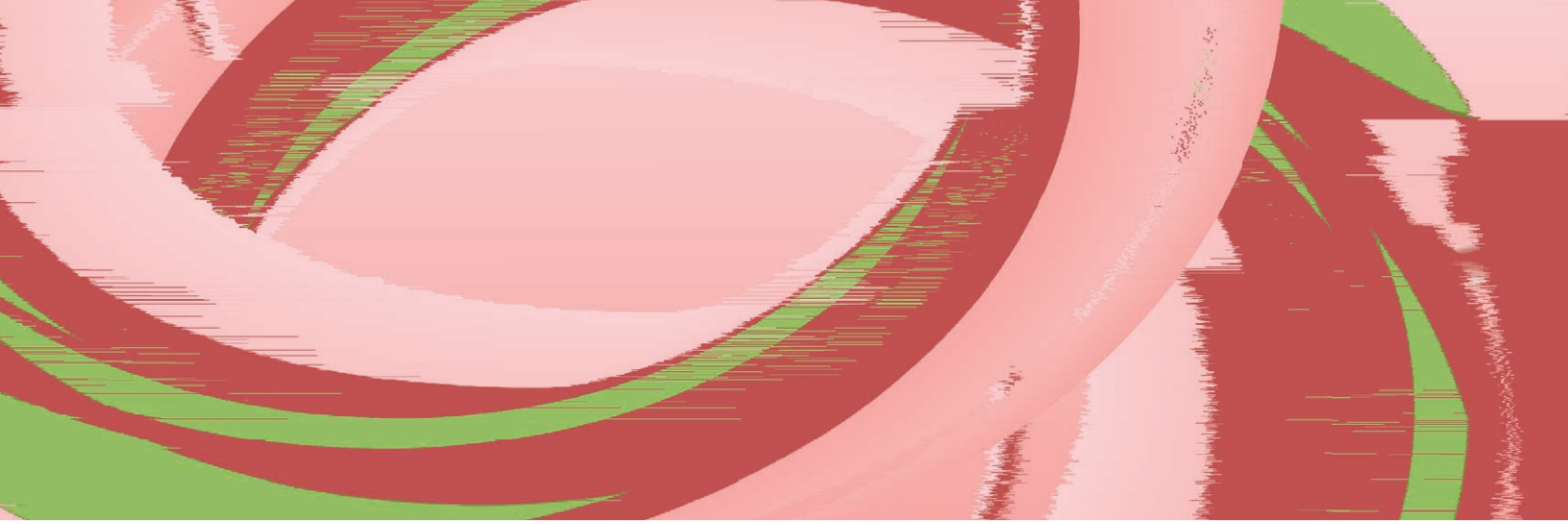
Research and development being performed at the Air Force Research Laboratory's (AFRL's) Information Directorate is leading the way by providing a scalable, reconfigurable, embedded processing capability to ingest raw sensor data, process the data into information, and distribute the products to the war fighter. The Information Directorate's work is being performed under two projects, Airborne Wide Area SAR Processing (AirWASP) and Radiofrequency Adaptive Persistent ISR Data (RAPID) Link. The AirWASP project is focused on the processing capability, and the RAPID Link project is developing the data air-to-ground downlink portion. This paper addresses the requirements, generation, processing needs, and system design for AirWASP.

For demonstration purposes, AirWASP has teamed with AFRL's Sensors Directorate and its Gotcha II program. Gotcha II is the Sensors Directorate's second generation of an ISR system that provides wide-area persistent surveillance using two synthetic aperture radar (SAR) systems. Such SAR systems are computational and data intensive, with large incoming (radar phase history) and outgoing (surveillance products produced) data streams. This teaming arrangement has been ongoing for a number of years, enabling each directorate to increase its system's capabilities with advancing technology. Increased capabilities, while beneficial when considering the outcome, poses a major challenge in terms of processing requirements, which are amplified when requiring processing to be done in real time and close to the sensor.

This particular ISR mission is considered a massive data problem as it consists of processing multiple channels of radar data arriving at 2 GB/s per channel to produce high-resolution SAR imagery for a 20-km-diameter area with 314-gigapixel images produced once every 2 s. When evaluating the processing power required to handle this amount of data, which is described in greater detail later, we see that over 80 trillion floating-point operations per second (Tflop/s) must be processed onboard an aircraft. To put into perspective the magnitude of this value, consider that one central processing unit (CPU) used to build the AirWASP system is capable of achieving 124.8 billion floating-point operations per second (Gflop/s) [2]. If using only the CPUs mentioned, hundreds of processors would be required, clearly eliminating any chance of processing onboard. While heterogeneous computing, which is used for this system, reduces the need for using so many CPUs, the ease of programming heterogeneous computers for optimal performance escalates the difficulty of the problem. Typically, real-time processing for massive data problems of this magnitude is done using large, power-hungry supercomputers located on the ground, with ample resources.

The airborne processing capability developed for the current iteration of AirWASP demonstrates substantial improvement over previously demonstrated capabilities [3]. The heterogeneous AirWASP high-performance embedded computer (HPEC) system, Coeus, was developed using state-of-the-art, general-purpose graphical processing units (GPGPUs) and CPUs—all while considering stringent size, weight, and power (SWaP) constraints. In the following sec-

Authors' current address: C. Usmail, M. Little, High Performance Systems, Information Directorate, Air Force Research Laboratory, 525 Brooks Road, Rome, NY 13441, USA, E-mail: courtney.usmail@rl.af.mil. R. E. Zuber, SRC, Inc., Syracuse, NY 13212, USA. Manuscript received April 11, 2013, revised August 13, 2013, and ready for publication October 17, 2013. Review handled by B. Rigling.
0885/8985/13/ $26.00 © 2014 IEEE



tions, we review the AirWASP project and cover requirements, generation, processing needs, tradeoffs, system design, and our future vision.

BACKGROUND

Embedded computing systems fulfill a range of mission needs and can be used in a variety of situations that are constrained by SWaP. The development of embedded systems has diversified to include meeting changing requirements [4]. Due in part to the drive for development of HPC systems for military ISR missions, the embedded computing industry has invested increasing resources toward developing HPEC systems [1]. For example, without the use of an HPEC system, a typical mission collecting 100 TB of sensor data could process and analyze only 0.09% of that data with current downlink capabilities [1]. The need for processing onboard and close to the sensor is increasingly prevalent and necessary in military applications.

Addressing the need for HPC while maintaining energy efficiency introduces a challenge in design, as greater computing power generally results in greater energy consumption [4]. The challenge in reducing power requirements for high-performance computers can be put into perspective when evaluating the peak power requirements for supercomputers on the Top500 List and Green500 List. Among those at the top of the list, several computers have a peak power performance requirement that rises above 10 MW, comparable to the amount of power required for a small city with 40,000 people [4]. By comparison, the typical power available for an embedded computer on a military system is a very small fraction of that. However, the processing requirements for many military problems remain large, exceeding the billion floating-point operations per second range.

AIRWASP HISTORY

Leveraging commercial off-the-shelf (COTS) systems in development of HPECs has been essential in reduction of cost and schedule during the development cycle; furthermore, it makes performing technical refresh or system upgrades easier [1]. The progression of HPEC systems for processing radar data at the Information Directorate has improved over time, as illustrated in **Figure 1**.

Figure 1 depicts the Gotcha SAR-type program development in the AFRL Information Directorate progressing from Swathbuckler, to VideoSAR, to AirWASP. Swathbuckler, which used AFRL's Coyote cluster and was completed in 2006, was designed for processing a swath of 40 km of high-speed radar returns using stripmap imagery. The architecture was composed of an analog-to-digital converter sampling at 2 GHz and a field-programmable gate array (FPGA)–based downconversion to baseband. An HPEC system with embedded software for real-time image formation and a control processor were also included to manage parallel processing activities. Cluster nodes were composed of dual Intel Xeon servers equipped with an FPGA board in the front [5]. The system performance requirement of 100 Gflop/s, sustained, was surpassed with the Coyote Cluster [6]. In addition, the use of COTS architecture allowed cost to remain low, especially when compared with other HPEC systems developed in the same time frame.

Completed in 2010, the VideoSAR project, using AFRL's Horus Cluster, is composed of real-time, spotlight SAR for a 9-km spot. As is the trend with government computing systems, employing COTS hardware helped keep costs low and facilitate changes and upgrades to the system. The Horus Cluster is composed of one head node, which provides input and output (I/O) services, and eight compute nodes, each consisting of a Dell server with two Nvidia Tesla C2050 GPGPUs [3]. The performance requirement of 22 Tflop/s for the VideoSAR project was significantly greater than that of Swathbuckler, at 100 Gflop/s [3], [6]. VideoSAR and AirWASP, while scalable and versatile, were designed with the Gotcha system in mind as a demonstration vehicle. As the Gotcha project progressed and the amount of data requiring processing commensurately and significantly increased, the need for improved HPEC system performance continued to rise as well, all while maintaining strict SWaP constraints.

DESIGNING THE SYSTEM

AirWASP is scalable across a range of requirements. However, for design purposes, the performance parameters of

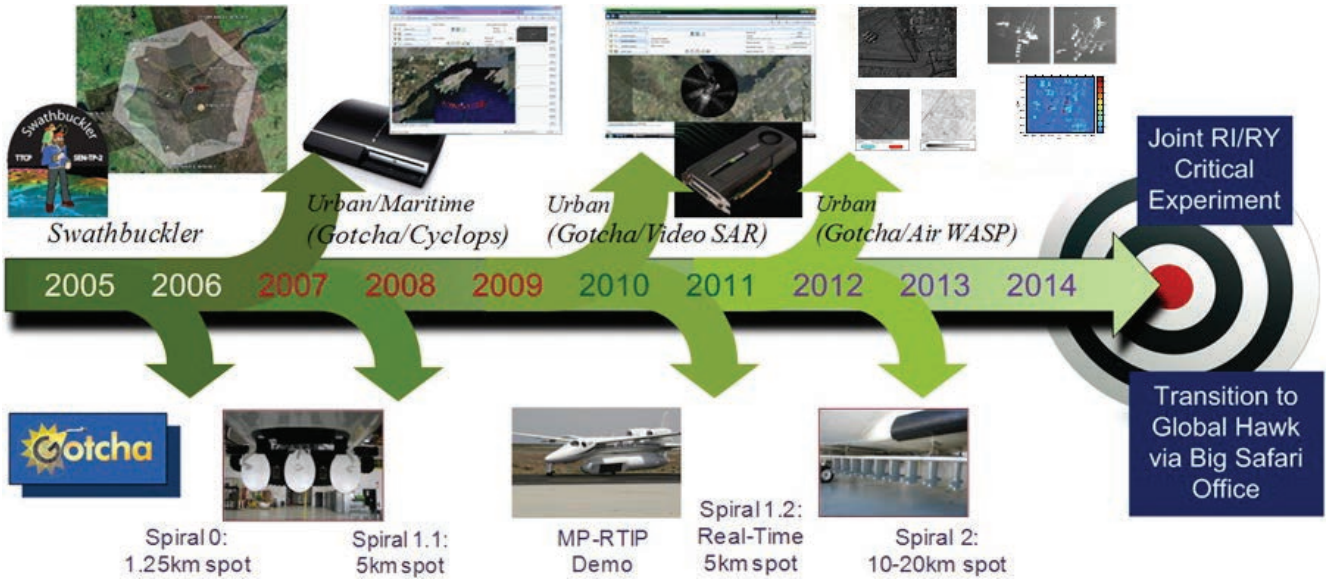


Figure 1.
Technology progression of embedded processing for onboard ISR systems.

the chosen demonstration vehicle, i.e., the Gotcha II system, were used (**Table 1**). As shown, the Gotcha II program has both an objective goal and a threshold goal. The AirWASP system was designed to meet the threshold goal; scalability was also considered during development in order to fulfill future needs of the objective goal.

The area of coverage listed in Table 1 depicts the diameter of the area of interest. One mode of the Gotcha system regards flying a sensor in a circular path around a given area of interest as providing persistent surveillance of that area. This concept not only provides persistence but also allows for the collection of multiple look angles over a shorter period of time compared with more conventional spotlight SAR [7]. Multiple look angles are necessary in order to create an image [7]. Given the circular area of interest, the following equation was used to determine the number of pixels for each frame F, where r is the radius of the area and R is the resolution of the image:

$$F = \frac{\pi r^2}{R^2} \tag{1}$$

Using the threshold values from Table 1 and Equation 1, one frame occupies 314 gigapixels. The processing power, or floating-point operations per second, necessary for processing SAR images of this magnitude was the driving point of system development. Estimating the number of floating-point operations required is dependent on the algorithm used for image formation. SAR data processing is typically done using one of two classes of algorithms: fast Fourier transform (FFT) based and backprojection. While algorithms based on FFT minimize the computational load and processing time, they are less ideal when considering image processing from a moving platform. For instance, FFT algorithms make assumptions that the aircraft flies in a straight line and at a consistent velocity and therefore is lacking in real-time situations [8]. The backprojection algorithm eliminates these issues at the price of increased computational complex-

Table 1.

Key Gotcha II Performance Parameters		
Parameter	**Objective**	**Threshold**
Area of coverage	20-km diameter	10-km diameter
Surveillance area	314 km^2	78.5 km^2
X-band resolution	0.3 m	0.5 m
UHF resolution	1 m	3 m
Frame rate	4 frame/s	1 frame/2s

ity. There are many algorithms based on the backprojection method that trade levels of precision for simplification. Among the variations are fast backprojection and full backprojection, two algorithms used for image formation in the AirWASP project; these are discussed in subsequent sections.

A base estimate for the number of floating-point operations per second required for processing the image is determined using Equation 2, as subsequently shown. The Gotcha II program uses both X-band and ultra-high-frequency (UHF) radar, as illustrated in Table 1. X-band radar is significantly more complex to process and is used in a 6:2 ratio compared to UHF. Therefore, it was the primary specification used for determination of system requirements. The pulse repetition frequency (PRF) for X-band radar given by Gotcha II specifications covers a range of values; for computational estimation purposes, a median value of 15 kHz was used. The PRF for UHF is roughly one-third that of X-band. Using a frame size (total number of pixels) of 314 gigapixels and a frame rate (number of frames per second) of 0.5 frame/s, Equation 2 was used, where 35 is the number of operations required to process 1 pixel using the single-stage backprojection algorithm [9]. Equation 2 quantifies the performance P (in units of floating-point operations per second) required to sustain a frame rate with a particular PRF, frame size, and frame rate.

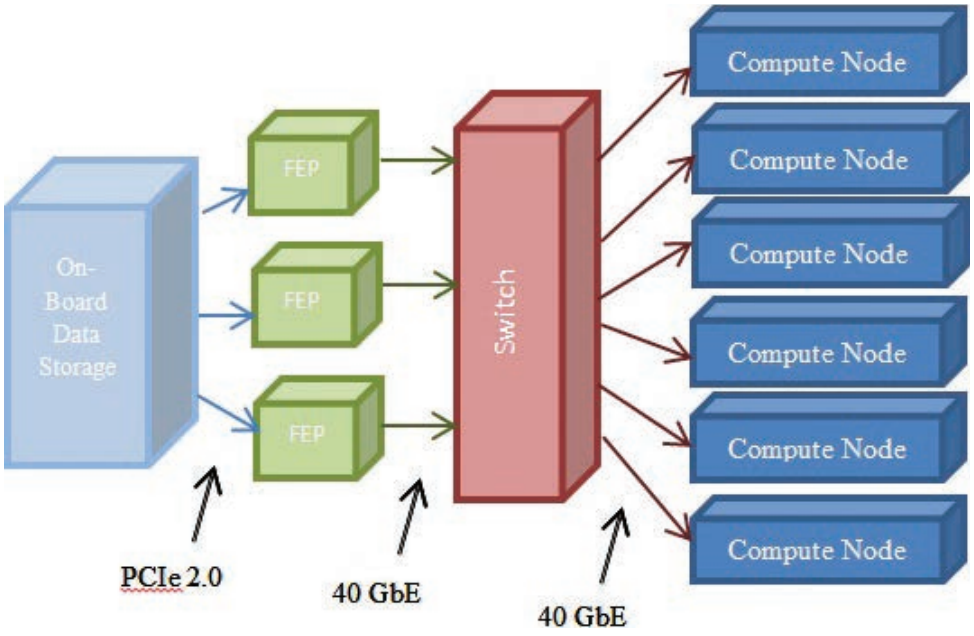


Figure 2.
Coeus block diagram.

$$P = 35 \times \mathrm{PRF} \times \frac{\text{Frame Size}}{\text{Frame Rate}} \tag{2}$$

For the 10-km-diameter spot case of Table 1, Equation 2 reveals that 82.5 Tflop/s sustained are required to process the wide-area SAR images in real time. The software development team leveraged backprojection algorithms that provided efficiency and reduced the overall processing performance requirements, having decreased the number of floating-point operations needed to produce a frame,

The computational requirements of the system, while demanding, are not a burden given current state-of-the-art computing architectures. The AirWASP system is data driven when processing in real time, which forms the main design and development difficulties. The AirWASP system receives data from an onboard storage system via seven peripheral component interconnect express (PCIe) 2.0 channels—six X-band channels and one UHF channel. The wire-speed transfer rate for PCIe 2.0 across 16 lanes is 8 GB/s, and the acquisition rate for the X-band data is 2 GB/s per channel, transferring about 720 GB/min of raw X-band data to the processor [10]. Front-end processors (FEPs) typically are used to assist in dealing with the high I/O data rates to reduce the likeliness of bottlenecks. A common architecture for front-end processing is FPGA-based and is of increasing interest in radar processing, as FPGAs excel in performing repetitive tasks quickly, as well as preparing digital signals for further processing [11]. PCIe connections as a source of data transfer also necessitates that the system include a switch for sharing data among compute nodes.

BUILDING THE SYSTEM

The AirWASP system is a heterogeneous system composed of state-of-the-art CPUs and GPGPUs that fulfill previously mentioned requirements. As shown in **Figure 2**, the system consists of FEPs, a 40-GbE switch (GbE is equal to 1-gigabit/s Ethernet, i.e., Ethernet capable of transmitting data at the line rate of 1 billion bit/s), and six compute nodes. The block diagram in Figure 2 demonstrates the framework for processing this particular problem; its versatility allows adjustment of the number of compute nodes to accommodate additional SWaP and mission constraints.

The data flow from the Gotcha II onboard processor requires that an FEP be included in the system, as previously mentioned. The FEP provides a 10-s buffer and does limited processing of data prior to distribution to the compute nodes.

The primary sources of computation for the AirWASP system are the six compute nodes, illustrated to the right in Figure 2. Each compute node consists of one Intel E-2690 high-performance server equipped with three Nvidia Tesla Kepler K20 GPGPUs. The use of GPGPUs for signal analy-

sis has become increasingly popular, as they diversify from high-level graphical imaging, such as that used in gaming, to general-purpose computing. GPGPUs are viable when parallelization of tasks is possible. Nvidia's Compute Unified Device Architecture programming model is especially useful in image processing with the use of the backprojection algorithm. as it takes full advantage of parallelization [12]. GPGPUs were also chosen as the primary compute source for the SAR data given the high computational requirement. This was first demonstrated with the VideoSAR project in 2010 (Figure 1) and is now being used in a somewhat different overall architecture in AirWASP. The most recent GPGPU on the market at the time of development, which was chosen for this project, was the Nvidia Tesla Kepler K20 [13]. The K20 provided a significant improvement over previous Nvidia GPGPU architectures used in AirWASP predecessors, such as the Nvidia C2050 GPGPUs used in the VideoSAR project, with a theoretical single-precision peak performance of 3.5 Tflop/s, far surpassing any CPU on the market [3], [13]. Each compute node is capable of achieving 13 Tflop/s single precision and consumes approximately 1000 W [13]. To stay within power constraints and meet the processing requirements, six compute nodes are used.

As mentioned, communication among compute nodes is necessary in processing the Gotcha II data. The 40-GbE switch allows for high-speed data transfer among nodes, increasing functionality of the system. The Dell Force10 Z9000 plug-and-play data center core switch offers a 32-port, 40-GbE switch and provides 2.5-Tb/s full-duplex, nonblocking line rate performance. The switch also supports 10-GbE connections if required for additional missions. The data switch consumes a maximum of 800 W [14].

As discussed earlier, onboard computing systems place a greater stress on the SWaP of the system. Based on the available space for the projected aircraft to be used by the Gotcha II program, SWaP constraints imposed on the AirWASP system are 27 rack units, 400 pounds, and 8000 W, respectively. While the AirWASP system will initially be flown on a manned aircraft, the flexibility to reorganize the framework will allow the system to be flown on more SWaP-constrained platforms, such as may be the case with unmanned aircraft.

SOFTWARE AND DATA PROCESSING

Instead of developing an information system and basis for a specific task, e.g., a particular surveillance product, the technology provides capabilities and options through a new and innovative framework that allows a rapid response to changing requirements—providing a truly robust and agile solution. Specifically, in our approach, hardware components such as CPUs and GPGPUs are linked to provide flexibility and enhanced operation through interlinked and configurable I/O connections and software components. Furthermore, each component is robust in that the configurations provide varied and multiple data I/O streams. The advantage to this component model is the configurability to reorganize the system's underlying operational framework in order to optimize processing-based needs and pertinent metainformation, which includes information on the raw data, the finished information products, and the system constraints, e.g., time, power, and storage. Therefore, the challenge addressed is understanding how to properly and effectively distribute each algorithm so as to minimize data movement and maximize processing throughput. These efforts expand the utilization of the code developed across multiple platforms, algorithms, system configurations, and ultimately, end-user needs.

To test these capabilities, a full backprojection SAR algorithm was implemented. Many SAR image formation implementations avoid this algorithm because of the computational burden. For an N by N pixel image, with N_p pulses of compressed range data, the number of operations is proportional to

$$\text{Operations} \approx N^2 N_p, \tag{3}$$

or on the order $O(N^3)$. While this puts an upper bound on the computational complexity, the value in pursing this algorithm is that it is processing intensive and applicable to various complex remote sensing and military applications. When considering a real-time case, if the processor is capable of backprojecting a single pulse of data ($N_p = 1$), to N by N pixels, at a rate less than or equal to one pulse repetition interval (PRI), the number of operations is on the order of $O(N^2)$ as determined by [3]. The total number of operations has not changed, although since we are processing in real time, we are constrained by the PRI. In exchange, the image formation is more robust, and it affords the opportunity to perform terrain height correction on the fly. In addition, for longer PRIs, assuming proper data support, techniques such as azimuth presuming may be employed to reduce the input data rate by downsampling azimuth data using a digital, low-pass filter [15]. These are specific examples of processing needs that are analyzed, as part of the framework, to aid in maximizing effective utilization of both the hardware and the software available.

In the following paragraphs, key components of the AirWASP design are discussed in greater detail. Emphasis is placed on the benefits of such an integrated software- and hardware-adaptable construct. The key components are composed of the following:

1. Hardware configuration and performance gains, which include effective use of memory, parallel processing, and compute node processing
2. Metainformation and parameters that were investigated as part of the use of this framework, i.e., options related to subapertures, coordinate systems, and pulses
3. Operational improvements through innovative use of range and pulse data framing and the importance of being able to process on various computing configurations while effectively managing data movement

Few assumptions are made about the underlying computational hardware, and it is up to the user to map the logical workflow of the processing to the available computing resources for maximum performance on a given system. The framework is composed of processing components that connect in a flexible manner to take advantage of the available hardware resources while trying to meet mission objectives. For example, the full backprojection SAR implementation is composed of several distinct components, each with a different function, including pulse preprocessing, pulse distribution, backprojection processing, subaperture summation, subimage integration, and VideoSAR. Each component has message passing interface connectors to pass data and control processing between compute nodes, as well as portable operating system interface threading connectors to enable parallel processing on each compute node. In addition, preallocated inbound and outbound round-robin memory buffers are used at all stages of communication. Data are received by each component, processed synchronously, and then asynchronously sent to the next component. Figure 3 characterizes the AirWASP framework from a software development point of view.

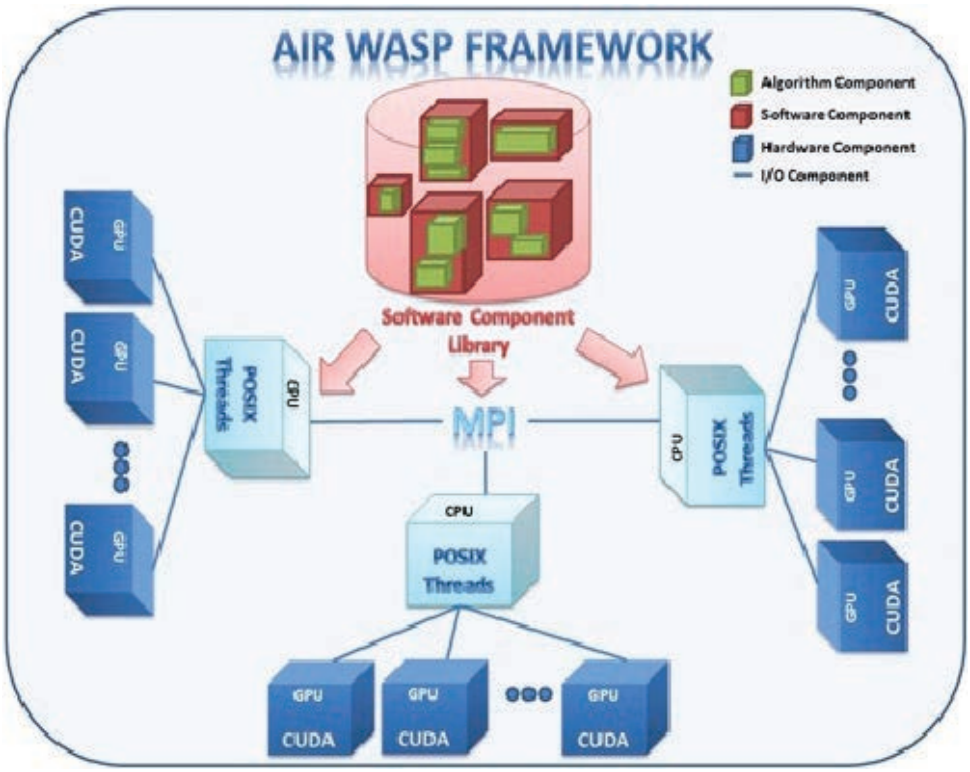


Figure 3.
The AirWASP framework [16].

Most SAR algorithms rely on precalculating coordinates in an Earth-centered, Earth-fixed coordinate frame or some other reference frame ahead of time. While this reduces the number of calculations needed for determining the range to each pixel, it has a hidden cost: coordinates must be loaded from global memory, which is relatively slow compared to processing speeds. In the case of AirWASP, all coordinates are calculated in an east, north, up (ENU) local reference frame relative to the image center. The lower-left coordinate is given some offset into the ENU plane. Since the pixels form a regular grid, the indices of the pixels plus the fixed offset (which can reside in cache) are used to calculate each geographic pixel location. When using terrain elevation data for height correction, the only additional piece of information that must be stored is the up component. As a result, less time is spent waiting on global memory reads. The object is to have the processors continuously processing data rather than idling because the next set of data is not available or falling behind because they cannot process the data fast enough.

Another optimization built into AirWASP is pulse range framing. In order to limit the amount of network and bus bandwidth needed, the software frames the data in the time domain, for each subimage, before being sent to a given backprojection component. In this manner, only the phase history data required by a given component are transferred, minimizing unnecessary data movement. This significantly eases the data rate requirements at any given node.

In addition, AirWASP contains an extensive and detailed logging subsystem that allows performance to be examined after each run. This allows for accurate pinpointing and examination of bottlenecks in the signal processing pipeline as both application software and system hardware evolve, with application software changing in terms of new functionality and hardware changing in terms of system-level optimizations, such as virtual interconnection network topologies. An example of an AirWASP configuration is illustrated in Figure 4.

Using 12 compute nodes of AFRL's Condor supercomputer (4 nodes are shown in Figure 4), each with two Nvidia C2070 GPUs (for a total of 16 GPUs), this architecture has demonstrated the ability to generate 3 × 3 km full backprojection VideoSAR imagery at 0.5-m resolution. This demonstration used a continuous stream of 2.6 million pulses at an input PRF of 3200 Hz for a total of 144 billion backprojections per second. Although some systems define their real-time capabilities based on their output rate, AirWASP depends on the input rate, which is driven by the PRF of the sensor. In other words, the goal of AirWASP is to process the input data, in this case range-compressed profiles, at the rate generated by the sensor (the PRF). Assuming a constant

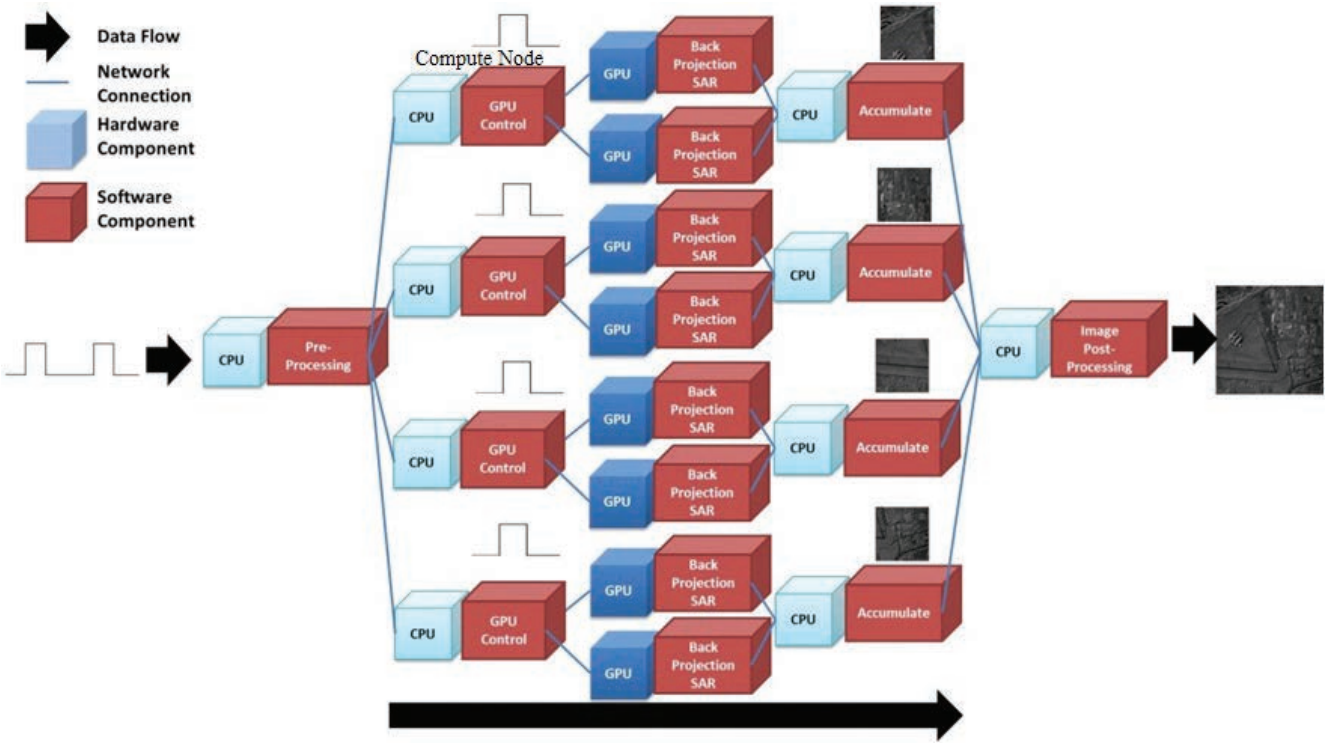


Figure 4.
Example of SAR imaging AirWASP topology [16].

synthetic aperture, the output data rate varies due to fluctuations in the sensor platform's speed. The image output rate was approximately 1.5 frame/s with less than 2 s of latency.

The AirWASP framework is designed to be expandable from its base circular spotlight VideoSAR mode. A coherent change detection (CCD) algorithm is being implemented, whereby a new CCD image is generated on each successive orbit. Parallel ground moving target indicator (GMTI) functionality is also planned. This functionality will work within the base architecture and only require the appropriate additional computing resources in order to accomplish the mission. In the case of CCD, these requirements are quite modest due to the slow update rate. For GMTI, a different phase center will be processed in parallel. This effectively doubles the input data rate. In order to accommodate the increased data rate for GMTI, the user can choose to either add more hardware to the system or reduce the size of the area processed.

CONCLUSION

The AirWASP framework leverages enabling technologies to aid in persistent surveillance missions onboard a manned aircraft. While the system as described was developed to fulfill the needs of the Gotcha II program, the system's versatility and modular design provide an abundance of technology options through framework design that will accommodate growing computing needs and can leverage technological advances. ◈

REFERENCES

[1] Perry, B. New demands on ISR require new technologies and systems. *Military Embedded Systems*, Vol. 7, 4 (June 2011).

[2] Advanced Clustering Technologies, Inc. Xeon E5 series overview, http://www.advancedclustering.com/downloads/.

[3] Barnell, M. Integration, Development and results of the 500 teraflop heterogeneous cluster (Condor). Presented at the HPC User Forum Meeting, San Diego, CA, Sept. 2011, http://www.hpcuserforum.com/presentations/Fall2011Presentations/SANDIEGO/MarkBarnellHPC-User-ForumCondor-8SEP2011.pdf.

[4] Munir, A., Ranka, S., and Gordon-Ross, A. High-performance energy-efficient multicore embedded computing. *IEEE Transactions on Parallel and Distributed Systems*, Vol. 23, 4 (Apr. 2012), 684–700.

[5] Linderman, R. Swathbuckler: Wide swath SAR system architecture. In *Proceedings of the IEEE Radar Conference*, Apr. 2006, 465–470.

[6] Linderman, R., and Tucker, S. Swathbuckler: Real-time wide swath synthetic aperture radar image formation using embedded HPC. In *Proceedings of the HPCMP Users Group Conference*, June 2006, 244–251.

[7] Stimson, G. W. *Airborne Radar.* Raleigh, NC: SciTech Publishing, 1998.

[8] Yegulalp, A. Fast backprojection algorithm for synthetic aperture radar. In *The Record of the IEEE Radar Conference*, 1999, 60–65.

[9] Majumder, U., and Soumekh, M. Tutorial T-18: SAR signal and image processing for ISR applications. In *The Record of the IEEE Radar Conference,* May 2010.

[10] PCI-SIG. PCI Express 2.0 Frequently Asked Questions, 2013, http://www.pcisig.com/news_room/faqs/pcie2.0_faq.

[11] Keller, J. Radar processing shifts to FPGAs and Altivec. *Military & Aerospace Electronics* (June 2003).

[12] Hartley, T., Fasih, A., Berdanier, C., Ozguner, F., and Catalyurek, U. Investigating the use of GPU-accelerated nodes for SAR image formation. In *Proceedings of the IEEE International Conference on Cluster Computing and Workshops,* Aug.–Sept. 2009, 1–8.

[13] Nvidia. Tesla Kepler GPU accelerators, http://www.nvidia.com/content/tesla/pdf/Tesla-KSeries-Overview-LR.pdf, 2012.

[14] Dell. Dell Force10 Z9000 data center core switch, http://i.dell.com/sites/doccontent/shared-content/data-sheets/en/Documents/Dell_Force10_Z9000_Spec_Sheet.pdf, 2012.

[15] Carrara, W. G., Goodman, R. S., and Majewski, R. M. *Spotlight Synthetic Aperture Radar: Signal Processing Algorithms.* Norwood, MA: Artech House, 1995.

[16] Zuber, R., Capraro, C., Barnell, M., and Little, M. Accelerated and agile exploitation framework leveraging GPUs. Presented at the Nvidia GPU Technology Conference, San Jose, CA, 2013, http://www.gputechconf.com/page/posters.html.

eaesm-29-01-05

1. Author: This article has been lightly edited for grammar, style, and usage. Please compare against your original document and make changes on these pages. Please limit your corrections to substantive changes that affect meaning. If no change is required in response to a question, please write "OK as set" in the margin. Copy editor
2. Author: “Radiofrequency Adaptive Persistent ISR Data (RAPID)” correct as edited? If not, please supply the correct full term for RAPID. Copy editor
3. Author: “Compute Unified Device Architecture” correct as edited? If not, please supply the correct full term for CUDA. Copy editor